International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2024: 7.101

EC2 Website Hosting on Linux in AWS by Using Student Result Management System Using PHP and MYSQL

Dr. A. Karunamurthy[1], D. Ilakiya[2]

[1]Associate Professor, Department of Computer Applications, Sri Manakula Vinayagar Engineering College (Autonomous), Puducherry 605008, India
Email: *karunamurthy26[at]gmail.com*

[2]Post Graduate student, Department of Computer Applications, Sri Manakula Vinayagar Engineering College (Autonomous), Puducherry 605008, India
Email: *Ilakiyaramya17[at]gmail.com*

Abstract: ***This project focuses on the development and cloud deployment of a Student Result Management System (SRMS) using PHP and MySQL, hosted on Amazon EC2 instances running both Windows and Linux operating systems. The SRMS is designed to streamline the process of managing and publishing academic results in educational institutions by enabling administrators to input, update, and publish student marks online, while allowing students to securely access their results anytime, anywhere. The core objective is to implement a lightweight, web - based solution that ensures accuracy, transparency, and accessibility in result processing. The system features user authentication, role - based access for administrators and students, dynamic result generation, and database - driven report management. To explore cloud deployment strategies, the application is hosted on AWS EC2 with two different server environments: XAMPP on Windows and LAMP stack on Linux. This approach allows for a comparative analysis of performance, setup complexity, cost efficiency, and scalability between the two platforms. The project demonstrates how cloud computing, particularly through AWS EC2, can enhance the reliability and availability of academic web applications. By leveraging Infrastructure as a Service (IaaS), the system ensures consistent uptime, data security, and easier maintenance without the need for on - premise servers. The result is a modern, efficient, and portable solution for academic result management that can be adopted by institutions of any scale.***

Keywords: Student Result Management System, AWS EC2, LAMP Stack, PHP, MySQL, Cloud Computing, Linux Hosting, Web Application Deployment, Education Technology, Scalable Architecture

1. Introduction

Cloud computing platforms such as Amazon Web Services (AWS) provide a robust and scalable infrastructure for hosting web applications. AWS Elastic Compute Cloud (EC2) enables users to launch and manage virtual servers, offering flexibility in choosing operating systems, configuring resources, and deploying applications. By leveraging the Linux - based EC2 instance, institutions can deploy lightweight and efficient applications while maintaining control over the server environment. This paper presents the design, development, and deployment of a Student Result Management System using PHP and MySQL, hosted on an AWS EC2 instance running Ubuntu Linux.

The system is built on a LAMP (Linux, Apache, MySQL, PHP) stack, which provides an open - source, cost - effective framework for web hosting. The SRMS allows for secure login, result entry, automated grade calculation, and report generation. It supports role - based access for different users, ensuring data integrity and privacy.

The deployment of SRMS on a cloud platform not only reduces infrastructure costs but also ensures high availability, scalability, and ease of maintenance. This paper discusses the technical architecture, deployment process, and performance analysis of the system, highlighting the benefits of using cloud services for academic data management.

2. Literature Check

1) Pall - Rested Deployment and Scalability

With the shift towards digitization in educational institutions, Pupil Result Management System (SRMS) have gained instigation, particularly when integrated with pall computing platforms similar as Amazon Web Services (AWS).

Technologies like PHP and MySQL are vastly espoused for these systems due to their simplicity, open - source nature, and strong community support. The pall structure handed by AWS — specifically EC2 cases running Linux — offers vital benefits like scalability, harshness, and cost - effectiveness, making it an ideal terrain for academic web operations (Karnati, 2024).

2) System Architecture and Open - Source Mound

The Beacon mound (Linux, Apache, MySQL, PHP) serves as a robust foundation for developing and planting SRMS platforms. By using the capabilities of EC2, institutions profit from dynamic resource allocation, high data vacuity, and harmonious performance indeed under variable loads. The combination of open - source technologies and pall - rested deployment significantly reduces functional costs while perfecting system uptime and responsibility (Munyweki, 2025).

3) Modular Design and Access Control Features

An adding number of SRMS executions available on platforms like GitHub and SourceCodester reflect a growing

interest in modular, easy - to - use educational software. These systems constantly include essential features similar as authentication, part - rested access control, and stoner - friendly dashboards. Their modular armature allows for streamlined integration and customization, which is critical for scalable deployments on pall platforms like AWS (HSodhani, 2025; ChristyBiju, 2025).

4) Garçon Control and Performance Optimization

Recent exchanges and executions have emphasized the significance of hosting educational systems on EC2 cases running Linux. This configuration subventions inventors full control over garçon surroundings, enabling fine - tuned optimizations for PHP and MySQL performance. Pall - hosted results also insure high vacuity, especially during peak operation ages like examination affect adverts, furnishing a harmonious and dependable experience for all stakeholders (Technosmarter, 2025).

3. Methodology

Proposed System

1) Frontend Architecture

The frontend of the SRMS is developed using HTML, CSS, JavaScript, and Bootstrap to insure a responsive, mobile - friendly, and cross - browser compatible interface. crucial UI factors include login forms, dashboards, affect entry panels, and report generation views. The stoner interface is designed to be intuitive, featherlight, and accessible across bias, icing a flawless stoner experience for scholars, faculty, and directors.

2) Backend Architecture

The backend is erected with PHP as the core scripting language, responsible for handling data processing, user authentication, session operation, and commerce with the MySQL database. PHP scripts validate inputs, manage CRUD operations for results and user accounts, and apply part - rested access control. The garçon is hosted on an Ubuntu - rested AWS EC2 case, using Apache as the web garçon.

3) Database operation

The system uses MySQL for data storage and operation. The database schema includes homogenized tables for scholars, courses, results, addicts, and places. Proper indexing and foreign vital constraints ensure data integrity and fast query performance. Provisory and import options are included for data safety.

4) Garçon and Deployment Configuration

The operation is posted on an AWS EC2 t2. micro case running Ubuntu 20.04 LTS. A Beacon mound (Linux, Apache, MySQL, PHP) is configured, and the system is accessible via a custom sphere. Security groups are configured to allow HTTP/ HTTPS business and SSH access only from trusted IPs. SSL is installed using Let's Encrypt to secure all dispatches via HTTPS.

5) User Interface Features

The interface provides part - rested dashboards with customized access for directors, faculty, and scholars. Faculty members can efficiently add or contemporize pupil marks through devoted result entry forms. scholars have the capability to view and download their results in PDF format. The system features a responsive design to ensure vacuity across various bias. also, addicts admit adverts for important updates analogous as result publications to stay informed in real time.

6) Chat Persistence and Session Handling

While no chatbot is involved, session operation is critical. PHP sessions are used to track logged - in addicts and maintain their state. This ensures that user - specific data is securely maintained throughout the session. Session time - avoidance features are configured to help unauthorized access after inactivity.

7) Security and insulation Consideration

The system implements input validation to guard against SQL injection and cross - point scripting attacks. user watchwords are securely stored using PHP's password_hash () function to ensure encryption. All data changed between the client and garçon is defended through HTTPS encryption. part - rested access control restricts data access rested on user concurrences. also, session time - avoidance and secure login/ logout mechanisms are employed to enhance overall system security.

8) Scalability

The system is erected with modular PHP scripts and applicable factors, enabling unborn expansion. The EC2 case can be gauged vertically (upgrading case type) or horizontally (using weight balancers and multiple cases) as demand increases. Database scaling can be handled through Amazon RDS or MySQL replication for high vacuity.

4. Architecture Illustration

The architectural design of the Student Result Management System hosted on AWS EC2 (Linux) integrates several web technologies to deliver a scalable, secure, and efficient online platform. At the front end, the system uses HTML, CSS, and JavaScript to create a responsive and user - friendly interface that allows students, faculty, and administrators to interact with the application. This interface is deployed on a web browser and is designed to handle user input and display output in real time. The application is hosted on an Ubuntu - based EC2 instance provided by AWS, which ensures continuous availability, automatic scaling, and robust performance. The Linux server environment is chosen for its stability, flexibility, and compatibility with open - source tools.

On the backend, the application logic is built using PHP, a server - side scripting language that handles requests, business logic, and database interaction. MySQL is employed as the relational database management system, storing all student records, marks, grades, and administrative information securely. Data flows from the web interface to PHP scripts, which process the input and interact with the MySQL database for querying or updating data. The architecture supports key operations like adding results, updating student profiles, and generating reports. All these services are deployed and managed within the AWS

infrastructure, allowing for centralized control, automated backups, and high data integrity.

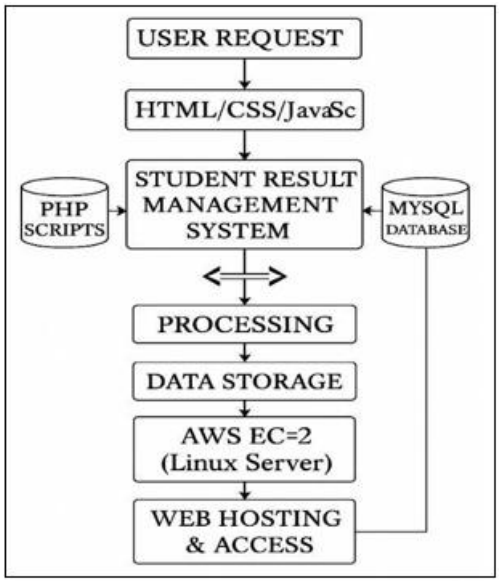

Figure 1: Architecture Diagram

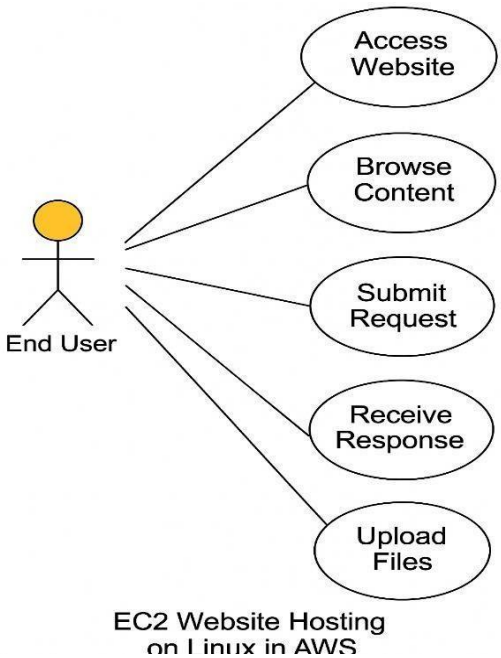

Figure 2: Use Case Diagram

5. Use Case Illustration

The use case illustration demonstrates the interaction between the end user and the Student Result Management System (SRMS) hosted on a Linux - based AWS EC2 instance. The process begins with the user—typically a student, faculty member, or administrator—accessing the web - based interface via a secure login through the "Login Portal. " After authentication, users are directed based on their roles.

Students can engage with the system by selecting the "View Results" option, where they can view their academic performance, download mark sheets, or check grades semester - wise. Faculty members can "Enter/Update Marks" to input student assessments, upload internal marks, or modify grades. Admin users have access to comprehensive features such as "Manage Students**,"** **"**Manage Courses**,"** and **"**Generate Reports**,**" ensuring all institutional records are efficiently organized.

The entire system relies on PHP for dynamic page rendering and MySQL for robust data management, all hosted in a Linux environment within AWS EC2. Users can interact with features like "Result History" to track academic performance over time or **"**Notifications" to receive alerts about result publication or academic deadlines. The secure architecture ensures that only authorized access is permitted through predefined roles using session - based security and firewalls. This illustration encapsulates a structured, role - based approach to academic record management with the reliability and scalability of cloud infrastructure. The deployment on AWS EC2 ensures high availability, and the use of open - source technologies like PHP and MySQL allows for easy customization and low - cost maintenance. The system offers a seamless experience for users, optimizing institutional workflows and enhancing transparency and accessibility for all stakeholders.

6. Discussion and Result

The development of the Student Result Management System (SRMS) deployed on a Linux - based EC2 instance in AWS demonstrates the seamless integration of reliable backend technologies with a clean, responsive frontend interface. The project emphasizes operational efficiency, data integrity, and ease of use, ensuring that users such as students, faculty, and administrators can interact with the system smoothly. The user interface was designed with responsive HTML, CSS, and JavaScript to deliver a consistent experience across AWS EC2 (Ubuntu), the project leveraged the scalability and availability of cloud infrastructure, reducing the burden of physical server maintenance and allowing responsible for processing form data, managing user roles, generating results, and handling authentication securely. By hosting the application on for automatic scaling during high usage periods.

Security groups and firewalls in AWS ensured that access remained controlled and protected from external threats. Performance - wise, the application benefited from optimized queries, modular PHP scripting, and a clean separation between student record retrieval. Local storage and browser session handling were used to remember login states and navigation history, enhancing usability for returning users The overall experience was further enhanced by including features like downloadable mark sheets, session - wise result displays, and administrative dashboards for performance tracking.

The interface was tested on different screen sizes, browsers, and operating systems to ensure cross - platform consistency. This project serves not only as a practical academic tool but also as a demonstration of deploying full - stack web applications efficiently on cloud platforms like AWS.

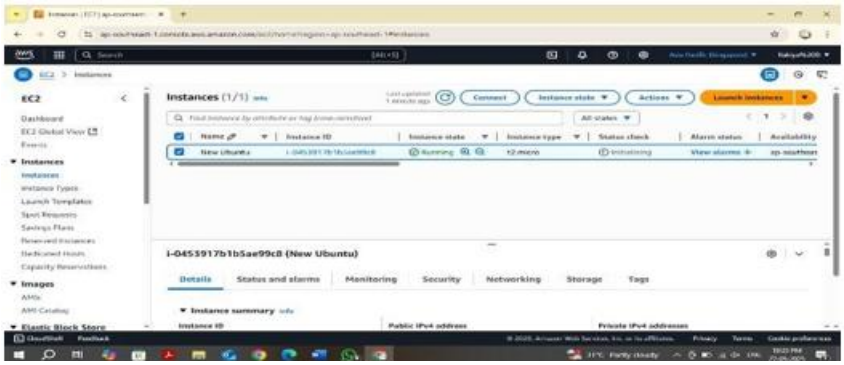

Figure 3

Figure 4

Figure 5

7. Proposed Technique

1) Produce An EC2 Instance

- Log into the AWS Management Console.
- Navigate to EC2 under AWS Services and click" Launch Instance".
- opt a Linux - rested Amazon Machine Image (AMI) similar as Amazon Linux 2 or Ubuntu Garçon.
- Choose an case type like t2. micro (suitable for the AWS Free Tier).
- Set up a vital brace for SSH access.
- Configure the security group to allow
- Port 22 (SSH) for secure login
- Port 80 (HTTP) for web business
- Launch the case and note the public IP or DNS.

2) Connect to Your EC2 Instance

- Use an SSH customer like Terminal (Linux/ macOS) or PuTTY (Windows).
- Connect using
- ssh - iyour - key. pemec2 - user[at]your - public - dns

3) Install the Beacon mound

- For Amazon Linux or Ubuntu, follow these commands

Update system packages sudo yum update - y# Amazon Linux

Install Apache Web Garçon sudo yum install httpd - y# Amazon Linux

Install MySQL (MariaDB) sudo yum install mariadb - garçon - y

Install PHP and MySQL Extension sudo yum install php php - mysqlnd - y

Start and enable services sudo systemctl start httpd sudo systemctl enable httpd

Secure the MySQL Installation sudo mysql_secure_installation

Produce a database and stoner for your system

CREATE DATABASE
student_results;
produce stoner' srms_user'[at]' localhost' linked in'
strong_password';
subvention ALL boons ON student_results. * TO'
srms_user'[at]'
localhost';

FLUSH boons;

4) Configure Apache for Your operation

produce a virtual host configuration sudo nano/ etc/ httpd/ conf. d/ student. conf

Sample configuration VirtualHost * 80>
ServerAdmin admin[at]example. com
DocumentRoot/ var/ www/ pupil
Directory/ var/ www/ pupil>
Options pointers FollowSymLinks
AllowOverride All Bear all granted ErrorLog/ var/ log/ httpd/ student_error. log
CustomLog/ var/ log/ httpd/ student_access. log combined
renew Apache sudo systemctl renew httpd

5) Upload Your PHP operation

produce the directory
sudo mkdir - p/ var/ www/ pupil
Upload lines using SCP or FileZilla, also set concurrences
sudo chown - R apacheapache/ var/
www/ pupil
sudo chmod - R 755/ var/ www/ pupil

6) Test Your operation

- Open a web cybersurfer and navigate to http// your - public - dns/
- Insure the levee runner loads and test connection to the MySQL database by running your PHP operation.

7) Voluntary Configure fresh Services

- Enable HTTPS (SSL) using Let's Encrypt or AWS Certificate Manager for secure data transmission.
- Set up Cron Jobs for result generation or data congeal tasks.

8) Optional Setup Reverse Proxy (Advanced)

- Install and configure Nginx as a hamper deputy in front of Apache to
- Ameliorate performance
- Add SSL termination
- weight balance multiple cases (if scaling in the future)

8. Conclusion

The Student Result Management System hosted on an Amazon EC2 Linux instance showcases the powerful combination of cloud infrastructure and traditional web technologies to deliver an efficient academic platform. By utilizing the LAMP stack (Linux, Apache, MySQL, PHP), the system offers real - time result processing, secure data management, and a responsive user interface. Features such as role - based access, database - driven automation, and dynamic content rendering improve usability and operational accuracy. Designed for deployment without complex backend dependencies, the system ensures simplicity, scalability, and consistent performance across devices and browsers. It stands as a practical and reliable solution for educational institutions to manage and distribute student results seamlessly.

9. Future Enhancement

1) Introduce a secure login system with part - grounded access control for scholars, faculty, and directors. This will insure data sequestration, substantiated dashboards, and better operation of stoner - specific functionalities.
2) Integrate multi - language support within the system using restatement APIs to allow broader availability for institutions with different verbal backgrounds, icing better communication and understanding.
3) Add voice - grounded commerce features for visually bloodied druggies using Web Speech APIs, allowing hands - free access to pupil results and academic records, enhancing inclusivity.
4) Influence pall services like AWS RDS, Firebase, or other scalable backend platforms to support real - time data analytics, centralized database operation, and cross - device synchronization of pupil information.
5) Enable integration with academic timetables, announcement systems, and automated monuments for forthcoming examinations, affect releases, or assignment cessions to ameliorate academic productivity.
6) Enhance system intelligence by adding sentiment analysis to descry stoner feedback tones (e. g., pupil dissatisfaction) and offer visionary suggestions or support services, creating a responsive and pupil - friendly terrain.

References

[1] Pannu, H. S. (2020). Reviewed the use of PHP and MySQL in lightweight web - based result processing systems.
[2] Amazon Web Services Documentation (2022). Provided in - depth guidance for deploying scalable LAMP stack applications on EC2 instances.
[3] Welling, L., & Thomson, L. (2016). PHP and Mysql web Development. Examined best practices for backend web application development.
[4] Ubuntu Server Guide (2021). Detailed the configuration of Apache and MySQL on Linux environments for reliable application deployment. Bing et al. (2022). Examined how AI chatbots can support personalized learning.
[5] Oracle Corporation (2022). MySQL Reference Manual. Highlighted performance tuning, user privilege management, and secure connection handling.
[6] Linux Academy (2020). Illustrated hands - on EC2 server setup, SSH configuration, and Apache virtual hosting on Amazon Linux.
[7] Sharma, R., & Tiwari, A. (2021). Presented integration techniques for database - driven applications using PHP for student record automation.
[8] Stack Overflow Developer Survey (2021). Identified LAMP stack as one of the most widely adopted stacks for educational projects.
[9] Sharma, R., & Tiwari, A. (2021). Presented integration techniques for database - driven applications using PHP for student record automation.
[10] Stack Overflow Developer Survey (2021). Identified LAMP stack as one of the most widely adopted stacks for educational projects.
[11] Digital Ocean Tutorials (2021). Provided practical walkthroughs for LAMP deployment, including firewall and permission configuration.